

CLAIMS

What is claimed is:

- 1 1. A method comprising:
 - 2 mapping a plurality of exception masks in a source architecture to an
 - 3 exception mask in a target architecture;
 - 4 executing translated code in the target architecture, the translated code
 - 5 representing binary code in the source architecture; and
 - 6 determining a state for the source architecture if an exception is raised while
 - 7 executing the translated code.
- 1 2. The method of claim 1, wherein determining a state comprises:
 - 2 determining an excepted instruction in the source architecture corresponding to
 - 3 the translated code that raised the exception;
 - 4 restoring the state of the source architecture to a pre-instruction state;
 - 5 masking all exceptions in the exception mask in the target architecture;
 - 6 re-executing the translated code corresponding to the excepted instruction; and
 - 7 analyzing a result of re-executing the translated code.
- 1 3. The method of claim 2, wherein analyzing a result comprises:
 - 2 determining an exception type for the exception that was raised; and
 - 3 examining the exception mask in the source architecture associated with the
 - 4 excepted instruction.
- 1 4. The method of claim 1 further comprising:
 - 2 selecting the state for the source architecture based on an exception type if the
 - 3 exception is genuine.

1 5. The method of claim 1 further comprising:
2 selecting a post-instruction state for the source architecture if the exception is
3 erroneous.

1 6. The method of claim 1, wherein mapping a plurality of exception masks
2 comprises:
3 performing a logical AND operation on the plurality of exception masks.

1 7. The method of claim 1 further comprising:
2 determining a translated code block to use as the translated code based on the
3 exception mask in the target architecture.

1 8. The method of claim 7 further comprising:
2 checking the exception mask against a masking assumption for the translated
3 code block.

1 9. The method of claim 8 further comprising:
2 generating two translated code blocks, an optimized code block with a
3 masking assumption that all exceptions are masked and a conservative code block
4 with a masking assumption that an exception is unmasked.

1 10. The method of claim 7, wherein determining a translated code block
2 comprises:
3 recognizing when the exception mask in the target architecture is changed by
4 an instruction in the source architecture.

1 11. A machine-readable medium providing instructions, which when executed by
2 a processing unit, causes the processing unit to perform operations comprising:
3 mapping a plurality of exception masks in a source architecture to an
4 exception mask in a target architecture;
5 executing translated code in the target architecture, the translated code
6 representing binary code in the source architecture; and
7 determining a state for the source architecture if an exception is raised while
8 executing the translated code.

1 12. The machine-readable medium of claim 11, wherein determining a state
2 comprises:
3 determining an excepted instruction in the source architecture corresponding to
4 the translated code that raised the exception;
5 restoring the state of the source architecture to a pre-instruction state;
6 masking all exceptions in the exception mask in the target architecture;
7 re-executing the translated code corresponding to the excepted instruction; and
8 analyzing a result of re-executing the translated code.

1 13. The machine-readable medium of claim 12, wherein analyzing a result further
2 comprises:
3 determining an exception type for the exception that was raised; and
4 examining the exception mask in the source architecture associated with the
5 excepted instruction.

1 14. The machine-readable medium of claim 11 further comprising:
2 selecting the state for the source architecture based on an exception type if the
3 exception is genuine.

1 15. The machine-readable medium of claim 11 further comprising:
2 selecting a post-instruction state for the source architecture if the exception is
3 erroneous.

1 16. The machine-readable medium of claim 11, wherein mapping a plurality of
2 exception masks comprises:
3 . performing a logical AND operation on the plurality of exception masks.

1 17. The machine-readable medium of claim 11 further comprising:
2 determining a translated code block to use as the translated code based on the
3 exception mask in the target architecture.

1 18. The machine-readable medium of claim 17 further comprising:
2 checking the exception mask against a masking assumption for the translated
3 code block.

1 19. The machine-readable medium of claim 18 further comprising:
2 generating two translated code blocks, an optimized code block with a
3 masking assumption that all exceptions are masked and a conservative code block
4 with a masking assumption that an exception is unmasked.

1 20. The machine-readable medium of claim 17, wherein determining a translated
2 code block comprises:
3 recognizing when the exception mask in the target architecture is changed by
4 an instruction in the source architecture.

1 21. An apparatus comprising:
2 a processing unit coupled to a memory through a bus; and

3 a binary translation process executed from the memory by the processing unit
4 to cause the processing unit to map a plurality of exception masks in a source
5 architecture to an exception mask associated with the processing unit, execute
6 translated code representing binary code in the source architecture, and determine a
7 state for the source architecture if an exception is raised while executing the translated
8 code.

1 22. The apparatus of claim 21, wherein the binary translation process further
2 causes the processing unit to determine an excepted instruction in the source
3 architecture corresponding to the translated code that raised the exception, restore the
4 state of the source architecture to a pre-instruction state, mask all exceptions in the
5 exception mask associated with the processing unit, re-execute the translated code
6 corresponding to the excepted instruction, and analyze a result of re-executing the
7 translated code to determine a state for the source architecture.

1 23. The apparatus of claim 14, wherein the binary translation process further
2 causes the processing unit to determine an exception type for the exception that was
3 raised, and examine the exception mask in the source architecture associated with the
4 excepted instruction to analyze a result.

1 24. The apparatus of claim 21, wherein the binary translation process further
2 causes the processing unit to select the state for the source architecture based on an
3 exception type if the exception is genuine.

1 25. The apparatus of claim 21, wherein the binary translation process further
2 causes the processing unit to select a post-instruction state for the source architecture
3 if the exception is erroneous.

1 26. The apparatus of claim 21, wherein the binary translation process further
2 causes the processing unit to perform a logical AND operation on the plurality of
3 exception masks to map the plurality of exception masks to the exception mask
4 associated with the processing unit.

1 27. The apparatus of claim 21, wherein the binary translation process further
2 causes the processing unit to determine a translated code block to use as the translated
3 code based on the exception mask in the target architecture.

1 28. The apparatus of claim 27, wherein the binary translation process further
2 causes the processing unit to check the exception mask against a masking assumption
3 for the translated code block.

1 29. The apparatus of claim 28, wherein the binary translation process further
2 causes the processing unit to generate two translated code blocks, an optimized code
3 block with a masking assumption that all exceptions are masked and a conservative
4 code block with a masking assumption that an exception is unmasked.

1 30. The apparatus of claim 27, wherein the binary translation process further
2 causes the processing unit to recognize when the exception mask associated with the
3 processing unit is changed by an instruction in the source architecture to determine the
4 translated code block.